Distance Calculation Between a Point and a NURBS Surface

Eva Dyllong and Wolfram Luther

Abstract. In this paper, we consider the computation of an Euclidean shortest path between a point and a modelled curve or surface in three-dimensional space, which is one of the fundamental problems in robotics and many other areas. A new accurate algorithm for the distance-calculation between a point and a NURBS curve and its extension to the case of a point and a NURBS surface is presented. The algorithm consists of two steps, and is crucially based on appropriate projections and subdivision techniques. To solve a nonlinear polynomial system derived from the classical formulation of the distance problem, the well-known Newton-type algorithms or subdivision-based techniques first considered by Sherbrooke and Patrikalakis are used. Their modifications in conjunction with a low subdivision depth in the presented algorithms yield a verified enclosure of the solution.

§1. Introduction

The distance-calculation is an essential component of robot motion planning and control to steer the robot away from its surrounding obstacles or to work on a target surface. The obstacles may be polyhedral objects, quadratic surfaces, which include spherical and cylindrical surfaces or more general surface types like the non-uniform rational B-splines (NURBS). Most of the well-known algorithms in the fields of computational geometry, robotics and Computer Aided Design are focused on computing the distance between polyhedra, as the problem is easier to solve and the answer is sufficient for many problems. For example, if a *free-form* designed obstacle like a NURBS surface is located at a great distance from a moving robot, then it is sufficient in the next step to know the distance values from certain sensor points on the robot to the convex hull of the NURBS control points, which describes a convex polyhedron, instead of the more time-consuming and expensive computation of the exact distance values. But if the robot approaches an obstacle, more details are necessary, and fast and accurate algorithms for finding the nearest point on the NURBS curve or surface are highly recommended.

Curve and Surface Design: Saint-Malo 1999
Pierre-Jean Laurent, Paul Sablonnière, and Larry L. Schumaker (eds.), pp. 55–62.

ISBN 0-8265-1356-5.

There is an abundance of literature to calculate the distance between convex and non-convex objects. For convex polyhedra, a lot of algorithms exist [1,4]. Two well-known algorithms, the Gilbert method (GJK, [8]) and the algorithm of Canny and Lin (CL, [3]) present iterative solutions to the problem, i.e., both construct a sequence of pairs of proposed distance points which are then improved by gradient descent. Another wide field consists of algorithms for general, mainly convex objects [5,7,13]. But in particular for objects defined by NURBS, there are only a few contributions in the literature. In [2], Cameron and Turnbull focus on computing the distance between convex objects defined by NURBS curves or patches, for which the critical step is the evaluation of the support mapping. The method is based on the Gilbert-Foo algorithm [7] for general convex objects with adjustments of the termination criteria like the support mapping for NURBS, which uses derivatives of their basis functions and the Newton-Raphson method (NR solver) for finding the roots. An algorithm for the computation of stationary points of a squared distance function is presented in [11]. This problem is converted to n_e polynomial equations with n_e variables expressed in a tensor product Bernstein basis. The solution method uses subdivision relying on the convex hull property of Bernstein polynomials and minimization techniques.

In this paper we describe a new algorithm which consists of two steps, and is mainly based on accurate projections and subdivision techniques. In the first step, the NURBS curve is decomposed into rational Bézier segments. Then some evaluations of suitable scalar products decide on further subdivision of a rational Bézier segment. This subdivision is iterated until certain criteria are fulfilled. In addition, a composition of the new method together with the classical formulation of the distance problem based on the calculation of a solution of nonlinear polynomial systems is presented. The algorithm is extended to the case of a NURBS surface.

§2. Problem Formulation

A NURBS curve $\vec{C}(u)$ of degree p is a vector-valued function of one parameter defined by

$$\vec{C}(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u)\omega_i \vec{P}_i}{\sum_{i=0}^{n} N_{i,p}(u)\omega_i}, \qquad a \le u \le b,$$

where $\{\vec{P}_i = (x_i, y_i, z_i) \in \mathbb{R}^3\}_{i=0}^{n}$ are the control points, $\{\omega_i\}_{i=0}^{n}$ are the weights, and $\{N_{i,p}(u)\}_{i=0}^{n}$ are the pth-degree B-spline basis functions defined on the knot sequence $U = \{u_i\}_{i=0}^{n+p+1}$ with $\{u_i = a\}_{i=0}^{p}$ and $\{u_i = b\}_{i=n+1}^{n+p+1}$. Let $\vec{P}^\omega = (\omega x, \omega y, \omega z, \omega) = (x', y', z', \omega') \in \mathbb{R}^4$ and H be the perspective map given by

$$\vec{P} = H\{\vec{P}^\omega\} = H\{(x', y', z', \omega')\} = \begin{cases} (\frac{x'}{\omega'}, \frac{y'}{\omega'}, \frac{z'}{\omega'}), & \text{if } \omega' \neq 0, \\ (x', y', z'), & \text{if } \omega' = 0. \end{cases}$$

Applying H to the nonrational B-spline curve in homogeneous coordinates

$$\vec{C}^\omega(u) = \sum_{i=0}^{n} N_{i,p}(u)\vec{P}_i^\omega \tag{1}$$

yields the corresponding NURBS curve $\vec{C}(u)$, i.e., $\vec{C}(u) = H\{\vec{C}^{\omega}(u)\}$.

Similarly to the curve case, we define a NURBS surface using the tensor product scheme. Accordingly, a NURBS surface $\vec{S}(u,v)$ is a bivariate vector-valued piecewise rational function of the form

$$\vec{S}(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\omega_{i,j}\vec{P}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\omega_{i,j}}, \qquad a \le u,v \le b,$$

with the bidirectional control net $\vec{P}_{i,j}$, the weights $\omega_{i,j}$ and the B-spline basis functions $N_{i,p}(u)$ and $N_{j,q}(v)$.

For a given point $\vec{Q} \in \mathbb{R}^3$ we address the problem of finding a shortest straight line segment $[\vec{Q},\vec{D}]$ with $\vec{D} \in \vec{C}(u)$ or $\vec{D} \in \vec{S}(u,v)$, respectively. We assume that all weights of the rational curves and surfaces are positive, to ensure that the convex hull property holds.

§3. Distance Algorithm for a Point and a NURBS Curve

In this section, an efficient and accurate algorithm for distance calculation between a given point $\vec{Q}$ and a NURBS curve $\vec{C}(u)$ is presented, which is a kind of an adaptive system of solution methods. The extension to the case of a NURBS surface works analogously.

The algorithm consists of two steps. In the first step, the NURBS curve is decomposed into rational Bézier segments $\vec{C}_j(u)$, $j = 1,\ldots,n_p$, which can be realized once in the preparation phase. In [12] Piegl and Tiller present an efficient algorithm for computing the n_p Bézier segments using the homogeneous form given by (1). Thus, our task is to compute the distance between a point and a rational Bézier curve. After decomposition, the distances between $\vec{Q}$ and each endpoint of the rational Bézier segments $\vec{C}_j(u)$ are calculated, and the smallest value is stored as a first rough approximation to the distance value d. In the second step, the rational Bézier segments are processed gradually. Let $\vec{P}_{j,k}$, $k = 0,\ldots,p$, be the control points of the j-th Bézier segment $\vec{C}_j(u)$, $j \in \{1,\ldots,n_p\}$. Then, for each $\vec{P}_{j,k}$, $k = 1,\ldots,p-1$, the distance to the straight line supporting the line segment $l(\vec{P}_{j,0},\vec{P}_{j,p})$ between the endpoints $\vec{P}_{j,0}$ and $\vec{P}_{j,p}$ is calculated, and for each projection point $\vec{R}_{j,k}$, $k = 1,\ldots,p-1$, on the line, we test if $\vec{R}_{j,k}$ belongs to the line segment $l(\vec{P}_{j,0},\vec{P}_{j,p})$, using suitable scalar product evaluations. If $\vec{R}_{j,k} \notin l(\vec{P}_{j,0},\vec{P}_{j,p})$ for at least one $k \in \{1,\ldots,p-1\}$, then the Bézier segment is subdivided into two Bézier segments, for which the second step of the algorithm has to be started again. Otherwise, the algorithm tests whether the Bézier segment can be replaced by the line segment $l(\vec{P}_{j,0},\vec{P}_{j,p})$ using the theorem by Wang and Xu (see Sec. 3.2). If $\vec{C}_j(u)$ is nearly a straight line, with a given accuracy ε, then the distance between the point $\vec{Q}$ and the line segment $l(\vec{P}_{j,0},\vec{P}_{j,p})$ is calculated. The distance d is updated if a smaller value is found, and j is replaced by $j+1$. If $\vec{C}_j(u)$ is not nearly a line segment, the following scalar products

$$s_k := (\vec{P}_{j,1} - \vec{R}_{j,1}) \cdot (\vec{P}_{j,k} - \vec{R}_{j,1}) \quad \text{for} \quad k = 2,\ldots,p-1,$$

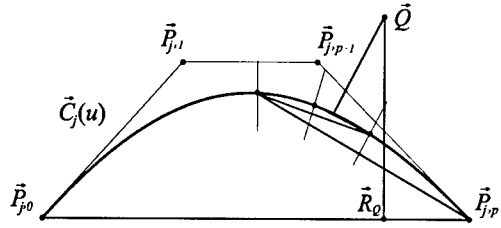

Fig. 1. Convergence of the algorithm.

are calculated. If all $s_k \geq 0$, i.e., all $\vec{P}_{j,k}$, $k = 1, \ldots, p-1$, are on the same side of the line supporting $l(\vec{P}_{j,0}, \vec{P}_{j,p})$, then the algorithm tests the position of point $\vec{Q}$. Otherwise, the Bézier segment $\vec{C}_j(u)$ is subdivided into two segments. To test whether $\vec{Q}$ lies in an influence area of the Bézier segment $\vec{C}_j(u)$, the projection point $\vec{R}_Q$ of $\vec{Q}$ on the line supporting $l(\vec{P}_{j,0}, \vec{P}_{j,p})$ is calculated and its position on the line is checked. If $\vec{R}_Q \notin l(\vec{P}_{j,0}, \vec{P}_{j,p})$, the distances $d(\vec{Q}, \vec{P}_{j,0})$ and $d(\vec{Q}, \vec{P}_{j,p})$ are calculated, d is updated if necessary and j is replaced by $j+1$. Otherwise, the Bézier segment has to be subdivided into two segments to increase the accuracy of the result. Fig. 1 shows how the algorithm works.

The subdivision of a rational Bézier segment can be continued until the termination criteria (see Sec. 3.2) are fulfilled or it can be interrupted after some steps, and afterwards the distance problem can be transformed in terms of the solution of the polynomial equation

$$(\vec{Q} - \vec{C}(u)) \cdot \vec{C}'(u) = 0 \tag{2}$$

in the variable u, where $\vec{C}'(u)$ describes the derivative of the curve $\vec{C}(u)$. This is mainly recommended for NURBS surfaces with a large curvature to avoid a high depth of subdivision. We calculate the roots of this equation using either the well-known (interval) Newton method, or one of the recently implemented solution methods briefly described in Sec. 3.3 (see [10]). A diagram illustrating the outline of the distance algorithms is given in Fig. 2.

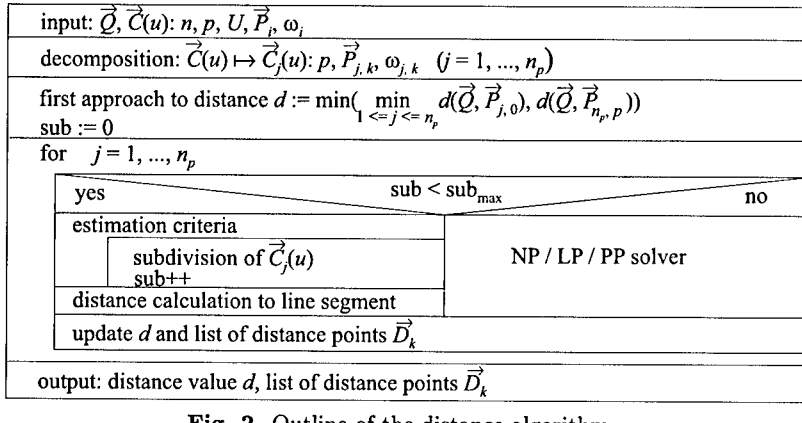

Fig. 2. Outline of the distance algorithm.

3.1. Subdivision

The subdivision of the control polygon of $\vec{C}_j^{\omega}(u)$ determinated by $\{\vec{P}_{j,k}^{\omega}\}_{k=0}^{p}$ into two Bézier segments with control points $\{\vec{Q}_{0,l}^{\omega}\}_{l=0}^{p}$ and $\{\vec{Q}_{1,l}^{\omega}\}_{l=0}^{p}$ works in homogeneous coordinates, and reads as follows:

```
for   k = 0 to p do
begin Q^ω_{0,k} := P^ω_{j,0}; Q^ω_{1,p-k} := P^ω_{j,p-k};
      for l = 0 to p-k-1 do
      P^ω_{j,l} := (P^ω_{j,l} + P^ω_{j,l+1})/2.0   end
```

Extending the idea from Bézier curve $\vec{C}_j^{\omega}(u)$ to Bézier surface $\vec{S}_{i,j}^{\omega}(u,v)$ by calling the routine twice, first in u and then in v direction, a subdivision of $\vec{S}_{i,j}^{\omega}(u,v)$ into four Bézier patches is realized (see [12]).

3.2. Termination criteria

In [15] Wang and Xu prove the following theorem:

Theorem 1. *For the rational Bézier curve $\vec{C}(u)$ of degree p,*

$$d(\vec{C}(u), l(\vec{P}_0, \vec{P}_p)) \leq \Phi(\omega_0, \ldots, \omega_p) \cdot \max_{1 \leq i \leq p-1} d(\vec{P}_i, l(\vec{P}_0, \vec{P}_p)),$$

where $\vec{C}(u) = \sum_{i=0}^{p} B_{i,p}(u)\omega_i \vec{P}_i / \sum_{i=0}^{p} B_{i,p}(u)\omega_i$ with the Bernstein polynomials $B_{i,p}(u)$ of degree p, and

$$\Phi(\omega_0, \ldots, \omega_p) := 1 - \left(1 + \max(\omega_0^{-1}, \omega_p^{-1})(\max_{1 \leq i \leq p-1} \omega_i)(2^{p-1} - 1)\right)^{-1},$$

$$d(\vec{C}(u), l(\vec{P}_0, \vec{P}_p)) := \sup_{a \leq u \leq b} \{\inf_{0 \leq t \leq 1} d(\vec{C}(u), t\vec{P}_0 + (1-t)\vec{P}_p)\}.$$

If after some subdivision steps, $d(\vec{Q}, l(\vec{P}_{j,0}, \vec{P}_{j,p})) \geq d + d(\vec{C}_j(u), l(\vec{P}_{j,0}, \vec{P}_{j,p}))$, or the curve can be approximated by $l(\vec{P}_{j,0}, \vec{P}_{j,p})$, i.e., $d(\vec{C}_j(u), l(\vec{P}_{j,0}, \vec{P}_{j,p}))$ is not greater than the desired tolerance ε, the subdivision of the segment stops after this step.

If $\omega_i = const$, the curve $\vec{C}_j(u)$ describes a Bézier curve, and the termination criterion of Theorem 1 is reduced to testing the following conditions: $(1 - 1/2^{p-1}) d(\vec{P}_{j,k}, l(\vec{P}_{j,0}, \vec{P}_{j,p})) < \varepsilon$ for $k = 1, \ldots, p-1$. In addition, for a Bézier curve the following theorem proved in [14] specifies the number of necessary subdivisions, i.e., after r subdivision steps the curve can be replaced by the line segments:

Theorem 2. *For the Bézier curve $\vec{C}(u)$ of degree p with control points $\{\vec{P}_i = (x_i, y_i, z_i)\}_{i=0}^{p}$ and any given $\varepsilon \geq 0$, let $L := \max_{0 \leq i \leq p-2}\{|x_i - 2x_{i+1} + x_{i+2}|, |y_i - 2y_{i+1} + y_{i+2}|, |z_i - 2z_{i+1} + z_{i+2}|\}$, and $r := \log_4(\sqrt{3}p(p-1)L/(8\varepsilon))$. Then, for $a \leq \alpha < \beta \leq b$ and $-\log_2((\beta - \alpha)/(b - a)) \geq r$,*

$$d(\vec{C}(u), l(\vec{C}(\alpha), \vec{C}(\beta))) \leq \varepsilon.$$

If at the beginning of the subdivision the value r can be calculated, then it is not necessary to perform the termination tests of Theorem 1.

3.3 Solution techniques and complexity analysis

The distance problem can be converted into a problem of computing all roots of a system of nonlinear polynomial equations in one or two variables. There are two techniques designed to solve such a problem in n_e variables efficiently; the projected-polyhedron (PP) and the linear programming (LP) technique, developed by Sherbrooke and Patrikalakis [10]. They rely on representation of polynomials in the multivariate Bernstein basis, the convex hull property and on the subdivision or linear programming technique. Alternatively, the Newton-Raphson method can be used to find the roots of the nonlinear polynomial equations.

Next, we analyse the amount of time required to execute each step of the distance algorithm in case of the NURBS curve. The decomposition of the curve $\vec{C}(u)$ of degree p into n_p Bézier segments takes at most $O(p \cdot n_p)$ operations, and the first approach to the distance value needs n_p+1 steps. The total cost of the distance calculation for n_p Bézier segments with subdivision depth of k is in worst case $O(2^k p^2 \cdot n_p)$ independently of the method used (subdivision-based technique or PP/LP solver) for finding the distance points. In this case ($n_e = 1$), close to a simple root, quadratic convergence is achieved.

§4. Distance Algorithms for a Point and a NURBS Surface

In the case of a NURBS surface $\vec{S}(u,v)$, the distance algorithm maintains its structure. After the decomposition of the surface into $n_p \cdot n_q$ Bézier patches $\vec{S}_{i,j}(u,v)$ with control points $\vec{P}^{i,j}_{k,l}$ ($0 \le k \le p$, $0 \le l \le q$), the first approximation to the distance value d is calculated taking the minimum of distances $d(\vec{Q}, \vec{P}^{i,j}_{k,l})$ for $k = 0, p$ and $l = 0, q$. The termination critera for subdivision of a non-degenerate Bézier patch $\vec{S}_{i,j}(u,v)$ ($\vec{P}^{i,j}_{0,0}$, $\vec{P}^{i,j}_{p,0}$, $\vec{P}^{i,j}_{0,q}$ are not collinear) are modified in the following way:

$$\Phi(\omega_{e,0},.,\omega_{e,q}) d(\vec{P}^{i,j}_{e,l}, l(\vec{P}^{i,j}_{e,0}, \vec{P}^{i,j}_{e,q})) \le \varepsilon,\ \Phi(\omega_{0,l},.,\omega_{p,l}) d(\vec{P}^{i,j}_{k,l}, l(\vec{P}^{i,j}_{0,l}, \vec{P}^{i,j}_{p,l})) \le \varepsilon$$
$$\text{and} \quad (\vec{P}^{i,j}_{p,q} - \vec{P}^{i,j}_{0,0}) \cdot \left((\vec{P}^{i,j}_{p,0} - \vec{P}^{i,j}_{0,0}) \times (\vec{P}^{i,j}_{0,q} - \vec{P}^{i,j}_{0,0}) \right) \le \varepsilon$$

for all $1 \le k \le p$, $1 \le l \le q$, and $e = 0, p$ (where $\times$ denotes the cross product). In this case $\vec{S}_{i,j}(u,v)$ can be replaced by a plane segment defined by $\vec{P}^{i,j}_{0,0}$, $\vec{P}^{i,j}_{p,0}$, and $\vec{P}^{i,j}_{0,q}$, the subdivision of $\vec{S}_{i,j}(u,v)$ is stopped, and the distance between the plane segment and the point $\vec{Q}$ is calculated. The remaining tests are performed in the u and v directions analogously to the curve case using the tensor-product structure of $\vec{S}_{i,j}(u,v)$. If the point $\vec{Q}$ does not lie in the influence area of the Bézier patch $\vec{S}_{i,j}(u,v)$, i.e., the projections onto the lines forming the boundary of the triangle defined by $\vec{P}^{i,j}_{0,0}$, $\vec{P}^{i,j}_{p,0}$, and $\vec{P}^{i,j}_{0,q}$ or defined by $\vec{P}^{i,j}_{p,q}$, $\vec{P}^{i,j}_{p,0}$ and $\vec{P}^{i,j}_{0,q}$ do not belong to sides of the triangle, the distance between the point $\vec{Q}$ and one of the boundary lines $\vec{S}_{i,j}(u,v)$, $u \in \{a,b\}$, $v \in \{c,d\}$, is calculated, d is updated if necessary, and the subdivision of $\vec{S}_{i,j}(u,v)$ is interrupted.

If a particular depth of subdivision is obtained, the subdivision of the Bézier patch can be stopped, and the PP or LP solver for nonlinear polynomial systems in two variables can be applied to the Bézier patch. The equations for $\vec{S}_{i,j}(u,v)$ read as follows analogously to (2):

$$\sum_{k=0}^{2p-1}\sum_{l=0}^{2q} a_{k,l}B_{k,2p-1}(u)B_{l,2q}(v) = 0 \text{ and } \sum_{k=0}^{2p}\sum_{l=0}^{2q-1} b_{k,l}B_{k,2p}(u)B_{l,2q-1}(v) = 0$$

with

$$a_{k,l} = \sum_{s=\max(0,k-p)}^{\min(p-1,k)} \sum_{t=\max(0,l-q)}^{\min(q,l)} \frac{\binom{p-1}{s}\binom{p}{k-s}\binom{q}{t}\binom{q}{l-t}}{\binom{2p-1}{k}\binom{2q}{l}}(\vec{P}^{i,j}_{s+1,t} - \vec{P}^{i,j}_{s,t})(\vec{P}^{i,j}_{k-s,l-t} - \vec{Q}),$$

$$b_{k,l} = \sum_{s=\max(0,k-p)}^{\min(p,k)} \sum_{t=\max(0,l-q)}^{\min(q-1,l)} \frac{\binom{p}{s}\binom{p}{k-s}\binom{q-1}{t}\binom{q}{l-t}}{\binom{2p}{k}\binom{2q-1}{l}}(\vec{P}^{i,j}_{s,t+1} - \vec{P}^{i,j}_{s,t})(\vec{P}^{i,j}_{k-s,l-t} - \vec{Q}).$$

The combination of a classical formulation of the distance problem and the subdivision technique is recommended if a high subdivision depth is expected, e.g., if r in Theorem 2 is too large in case of a very bent Bézier curve.

§5. Concluding Remarks

The method described in this paper computes the distance between a point and a NURBS curve or surface. Our goal was to provide a reliable method to solve this problem. The first few subdivision steps and tests quickly locate the regions of potential solutions. Then the subdivision can be either continued, or one of the equation solvers or even a distance-calculation algorithm for polyhedra can be applied [4,5]. We have developed an interval version of the PP/LP algorithm using interval arithmetic and considering a correct handling of roots of order two, suitable modifications of Graham's scan algorithm for building the convex hull, the revised simplex method by Gass, and an adapted interval-based subdivision by de Casteljau. The solver has been implemented in C++ using the library Profil/BIAS (see [9]). This improves the robustness of the distance-algorithm, assures an interval enclosure of the solution, and makes it suitable for verification of off-line tasks in path planning.

Some modifications to the algorithms could improve performance, e.g., if upper bounds on the derivatives of order two for the curve or surface are known [6]. But doubtless our NURBS-based algorithm will be slower, e.g., compared with our algorithms [4,5], where we deal with the objects as polyhedra. In a more complete distance tracking system, such as a manipulator in a complex environment, a progressive switch from a spherical or polyhedral enclosure of the objects to NURBS surfaces is recommended, especially if contact problems are investigated.

Acknowledgments. This research was supported by Deutsche Forschungsgemeinschaft within the scope of Sonderforschungsbereich 291 "Elastic manipulation systems for heavy loads in complex environments".

References

1. Bobrow, J. E., A direct minimization approach for obtaining the distance between convex polyhedra, Int. J. Robotics Research **3**, Vol. 8 (1989), 65–76.
2. Cameron, S. and C. Turnbull, Computing distances between NURBS-defined convex objects, IEEE Int. Conf. Robotics & Autom. (1998), Leuven, 3685-3691.
3. Canny, J. and M. Lin, A fast algorithm for incremential distance calculation, IEEE Int. Conf. Robotics & Autom. (1991), Sacramento, 1008–1014.
4. Dyllong, E., W. Luther, and W. Otten, An accurate distance-calculation algorithm for convex polyhedra, Reliable Computing **3**, Vol. 5 (1999), 241-254.
5. Dyllong, E. and W. Luther, An accurate computation of the distance between a point and a polyhedron, to appear in special issues of ZAMM (Proceedings of GAMM'99, April 12-16, Metz).
6. Filip, D., R. Magedson, and R. Markot, Surface algorithms using bounds on derivatives, CAGD **3** (1986), 295–311.
7. Gilbert, E. G. and C-K. Foo, Computing the distance between general convex objects in three-dimensional space, IEEE Trans. Robotics & Automation **1**, Vol. 6 (1990), 53–61.
8. Gilbert, E. G., D. W. Johnson, and S. Keerthi, A fast procedure for computing the distance between complex objects in three-dimensional space, IEEE J. of Robotics & Automation **2**, Vol. 4 (1988), 193–203.
9. Küppel, O., BIAS - Basic interval arithmetic subroutines, Technical report 93.3, TU Hamburg-Harburg, July 1993.
10. Patrikalakis, N. M. and E. C. Sherbrooke, Computation of the solutions of nonlinear polynomial systems, CAGD **10** (1993), 379–405.
11. Patrikalakis, N. M., E. C. Sherbrooke, and J. Zhou, Computation of stationary points of distance functions, Engineering with Computers **9** (1993), 231–246.
12. Piegl, L. and W. Tiller, *The NURBS book*, Springer, 1995.
13. Sato, Y. et al., Algorithms for convex and non-convex objects, in Proc. IEEE Int. Conf. Robotics & Automation (1996), Minneapolis, 771–778.
14. Wang, G., The subdivision method for finding the intersection between two Bézier curves or surfaces, Zhejiang University J., Special Issue on Computational Geometry 1984 (in Chinese).
15. Wang, G. and W. Xu, The termination criterion for subdivision of the rational Bézier curves, Graphical Models and Image Processing **1**, Vol. 52 (1991), 93-96.

Eva Dyllong and Wolfram Luther, Dept. of Computer Science,
University of Duisburg, D-47048 Duisburg, Germany
dyllong,luther@informatik.uni-duisburg.de